

СИСТЕМА ЗА КРЪВОДАРЯВАНЕ

LIFELOOD



КУРСОВ ПРОЕКТ

ДИСЦИПЛИНА: ПРОЕКТИРАНЕ И ИНТЕГРИРАНЕ НА СОФТУЕРНИ СИСТЕМИ

ФАЗА 2: АНАЛИЗ НА ИЗИСКВАНИЯТА И ПРОЕКТИРАНЕ НА СИСТЕМАТА

ВЕРСИЯ 1.0

ФАК. №	ИМЕ НА СТУДЕНТ	СЕКЦИЯ ОТ ДОКУМЕНТА
62113	Момчил Игнатов	АНАЛИЗ НА ИЗИСКВАНИЯ; ПОТРЕБИТЕЛСКИ ИНТЕРФЕЙС
62117	Гъокан Сюлейманов	МАТРИЦА НА ИЗИСКВАНИЯТА; ВЪВЕДЕНИЕ;
62137	Илкнур Мустафа	СИСТЕМНА АРХИТЕКТУРА; ДАННОВО ПРОЕКТИРАНЕ; КОМПОНЕНТНО ПРОЕКТИРАНЕ;

ДЕКАЕМВРИ, 2020

Съдържание

1	ВЪВЕДЕНИЕ	3
1.1	Цел	3
1.2	Обхват	3
1.3	Дефиниции и акроними	3
2	АНАЛИЗ НА ИЗИСКВАНИЯТА	4
2.1	Функционални изисквания	4
2.2	Нефункционални изисквания	5
2.2.1	Изисквания към потребителския интерфейс	5
2.2.2	Производителност	6
2.2.3	Наличност	6
2.2.4	Сигурност	6
2.2.5	Съответствие със стандарти	6
3	СИСТЕМНА АРХИТЕКТУРА	7
3.1	Архитектурно проектиране	7
3.2	Описание на декомпозицията	8
3.3	Обосновка на проектирането	10
4	ДАННОВО ПРОЕКТИРАНЕ	12
4.1	Описание на данните	12
4.2	Речник на данните	13
5	КОМПОНЕНТНО ПРОЕКТИРАНЕ	16
6	ПРОЕКТИРАНЕ НА ПОТРЕБИТЕЛСКИЯ ИНТЕРФЕЙС	16
6.1	Обобщение на потребителския интерфейс	16
6.2	Екранни изображения	17
7	МАТРИЦА НА ИЗИСКВАНИЯТА	21

1 ВЪВЕДЕНИЕ

1.1 Цел

Настоящият документ описва изискванията, архитектурата и процеса на проектиране на софтуерната система LifeBlood.

1.2 Обхват

В днешно време все по-често хората се нуждаят от кръвопреливане. Ситуацията налага реализирането на централно място, където хората лесно да заявяват своята готовност за кръводаряване, а центровете по кръводаряване лесно да обработват тези заявки. Системата представлява платформа за кръводаряване, която решава този проблем и свързва кръводарителите и центровете за кръводаряване.

1.3 Дефиниции и акроними

- **API (Application Programming Interface)** – приложно-програмен интерфейс
- **GDPR (General Data Protection Regulation)** – регламент за защита на личните данни, заведен като регламент (на Европейския парламент и на Съвета на Европейския съюз
- **REST (REpresentational State Transfer)** – стил софтуерна архитектура за реализация на уеб услуги
- **Бисквитки** - пакет информация, изпратен от уеб сървър към браузър, а след това връщан от браузъра всеки път, когато се достъпва до този сървър
- **Микросървис архитектура** – вариант на ориентираната към услуги архитектура, при която системата е разделена на няколко слабо свързани помежду си подсистеми (*микросървиси*)
- **Контейнер** - стандартна единица софтуер, който пакетира кода и всички негови зависимости, така че приложението да работи бързо и надеждно от една изчислителна среда в друга.

2 АНАЛИЗ НА ИЗИСКВАНИЯТА

Приоритет:

- Приоритет 1 – изискването е „задължително“
- Приоритет 2 – изискването е „необходимо“ за подобряване на софтуера
- Приоритет 3 – изискването е „препоръчително“

2.1 Функционални изисквания

Идентификатор	Изискване	Коментар	Приоритет
FR_01	Системата задължително трябва да поддържа регистрация на кръводарител.	Регистрацията включва предоставяне на лична информация като име, пол, възраст, кръвна група, както и задаване на имейл и парола.	1
FR_02	Системата задължително трябва да поддържа вход като кръводарител и като болничен служител.	Двата типа потребители имат различно ниво на достъп до функционалностите на системата.	1
FR_03	Системата задължително трябва да разрешава кръводарител да прави заявка за даряване.	При заявка, кръводарителят прави избор на център, където би желал да дари.	1
FR_04	Системата е необходимо да показва на кръводарител статус на заявките за даряване – текущи и предходни.		2
FR_05	Системата е препоръчително да предоставя възможност на болничен служител да търси по ключови думи заявки за кръводаряване.		3
FR_06	Системата е необходимо да показва на кръводарител карта с всички центрове за кръводаряване.	На картата, за всеки център се показва информация като телефон за връзка, адрес и работно време.	2
FR_07	Системата е препоръчително да показва на кръводарител и болничен служител статии и новини, свързани с темата за кръводаряване.		3
FR_08	Системата е задължително да показва на болничен служител		1

	списък с информация за всички заявки за кръводаряване.		
FR_09	Системата е необходимо да предоставя възможност на болничен служител да задава статуса на заявка за кръводаряване.		2
FR_10	Системата е препоръчително да предоставя възможност на болничен служител да търси по ключови думи заявки за кръводаряване.		3
FR_11	Системата е задължително да показва на болничен служител списък с информация за всички дарители.		1
FR_12	Системата е необходимо да предоставя възможност на болничен служител да редактира/изтрива данни за дарител.	Необходимо в случай на грешно въведени данни при регистрация от страна на дарителя.	2
FR_13	Системата е препоръчително да предоставя възможност на болничен служител да търси по ключови думи списъка с дарители.		3
FR_14	Системата е задължително да показва на болничен служител списък с информация за всички кръвонуждаещи се.		1
FR_15	Системата е необходимо да предоставя възможност на болничен служител да редактира/изтрива данни за пациент (кръвонуждаещ се).		2
FR_16	Системата е необходимо да предоставя възможност на болничен служител да регистрира нов пациент.		1
FR_17	Системата е препоръчително да предоставя възможност на болничен служител да търси по ключови думи списъка с пациенти.		3

2.2 Нефункционални изисквания

2.2.1 Изисквания към потребителския интерфейс

Потребителският интерфейс трябва да бъде удобен за работа от потребителите – навигацията между различните страници да бъде интуитивна и опростена.

При използване на устройства с различни пропорции и разделителна способност на экрана, потребителският интерфейс трябва да се адаптира спрямо тези особености (responsive interface).

2.2.2 Производителност

Системата трябва да поддържа едновременен достъп до 10,000 потребители и да зарежда уеб страницата за под 5 секунди в натоварен режим на работа.

2.2.3 Наличност

Системата трябва да бъде налична в 98% от времето, като е допустимо временно спиране на системата за поддръжка само в късните часове на денонощието.

2.2.4 Сигурност

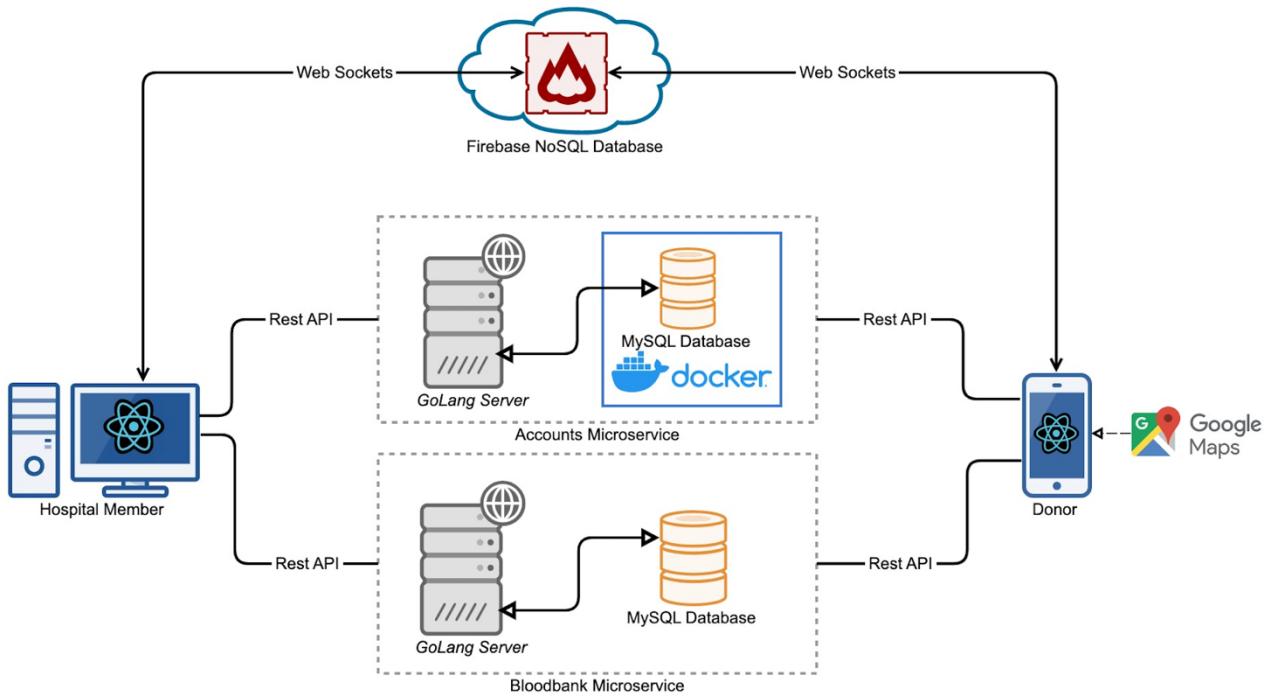
Системата трябва да използва сигурни протоколи за комуникация и данните да бъдат защита от неправомерен достъп.

2.2.5 Съответствие със стандарти

Системата трябва да бъде съвместима с GDPR стандартите за обработване на лични потребителски данни. Ако се използват "бисквитки", потребителят трябва да бъде уведомен чрез потребителския интерфейс.

3 СИСТЕМНА АРХИТЕКТУРА

3.1 Архитектурно проектиране



Системата е проектирана с помощта на 6 основни под-системи.

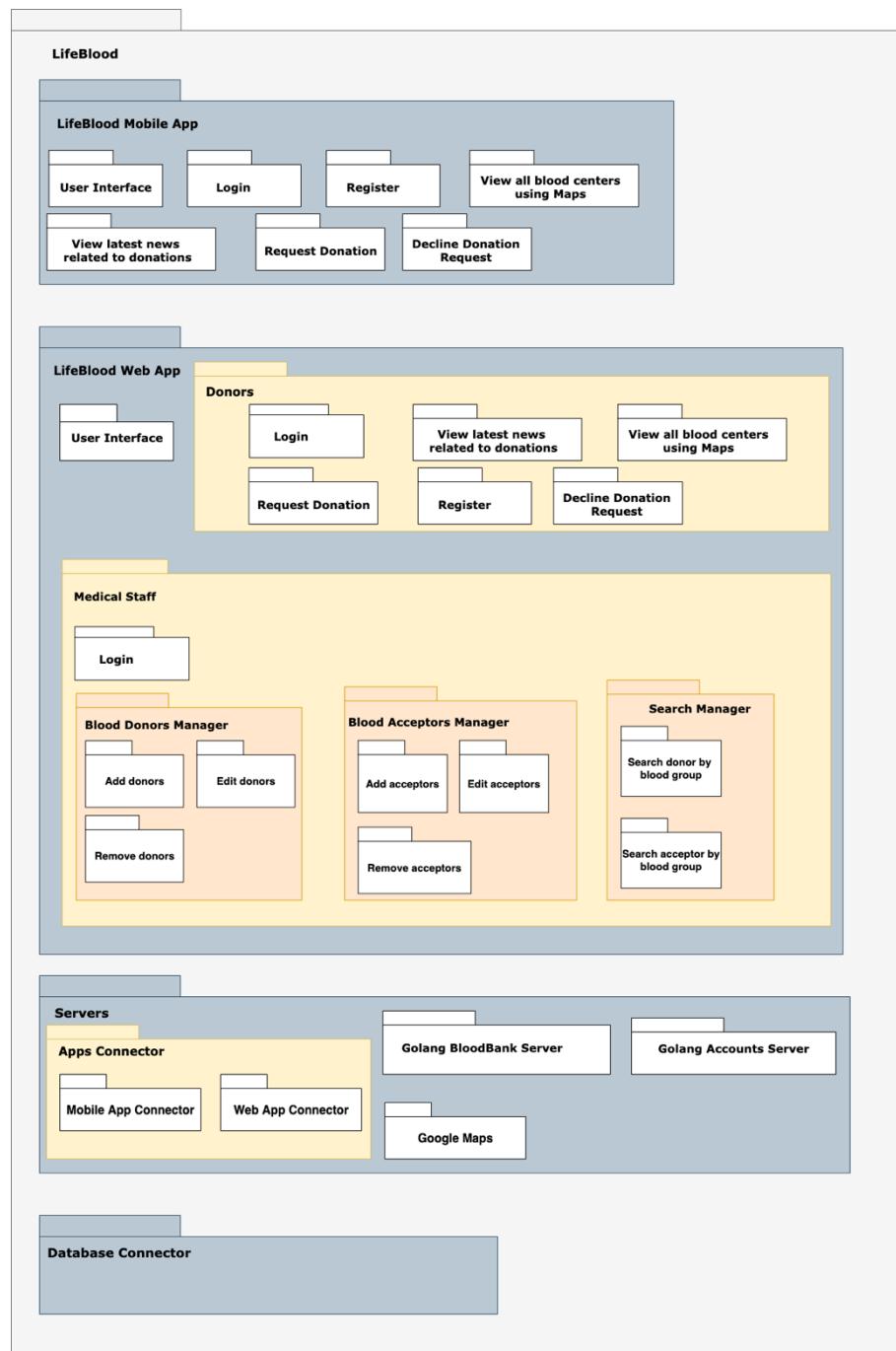
- Потребителски интерфейс - като болничен служител или донор
- NoSQL база съхраняваща в реално време информация за налични болници
- BloodBank Microservice
- MySQL база данни отговорна за операции в BloodBank подсистема
- Accounts Microservice
- Контейнеризирана MySQL база данни отговорна за операции в Accounts подсистема

Потребителския интерфейс създава Web Socket връзка към NoSql Firebase Realtime база от данни. Когато данните се обновяват на Firebase базата, потребителския интерфейс получава нови данните ако Web Socket връзката е отворена.

Потребителския интерфейс комуникира с microservices сървъри използвайки REST за комуникация. Различните microservices имат различен endpoint за да бъдат разграничавани в контекста на Потребителски интерфейс. Всеки microservice използва различна MySQL база от данни.

3.2 Описание на декомпозицията

Декомпозиция на системата на модули се представя по следния начин:



Подробно описание на модулите:

1. LifeBlood Mobile App

Предназначението на модула е да осигури мобилно приложение, което обикновените потребители (донори) да използват за изпращане на заявки за даряване на кръв до кръвни центрове.

Основни отговорности на модула са:

- Предоставяне на потребителски интерфейс със следните функционалности:
 - Регистрация и вход в системата
 - Начална страница, включваща:
 - Статии и новини за кръводаряване
 - Статистически данни за излекувани, центрове, донори
 - Търсене на центрове за кръводаряване
 - Управление на заявките за даряване на кръв на текущия потребител

2. LifeBlood Web App

Предназначението на модула е да осигури уеб приложение, което болничните служители да използват за връзка със системата.

Основни отговорности на модула са:

- Предоставяне на потребителски интерфейс със следните функционалности:
 - Вход в системата
 - Управление на заявките за даряване на кръв на всички потребители
 - Управление на данните на всички потребители
 - Управление на данните на всички пациенти
 - Добавяне на нов пациент

2.1 Подмодули

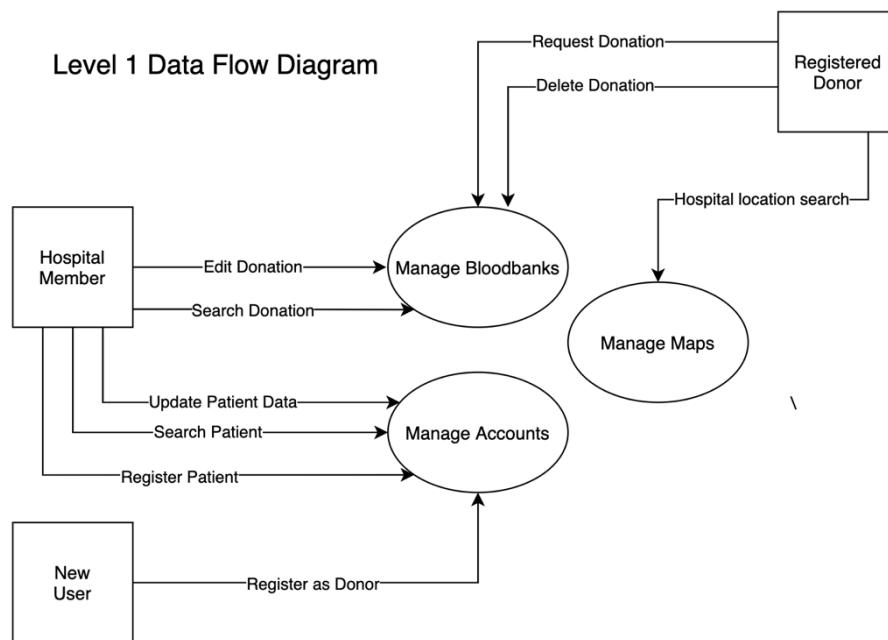
1. Донори - възможност за регистриране и вход в системата, търсене на кръвни центрове, управление на заявките за даряване на кръв.
2. Болнични служители - възможност за вход в системата, операции над донорите и пациентите, нуждаещи се от кръв в системата(изтриване, добавяне, редактиране), филтриране на пациенти и донори по кръвна група
3. Servers

Предназначението на модула е да осигури основната бизнес логика на системата и да осигури връзката между всички модули на системата.

Основни отговорности на модула са: Предоставяне на връзка с Google Maps, връзка с другите приложения на системата както и връзка с отделните микросървиси.

3.1 Подмодули

1. Google Maps
2. Apps Connector – осигурява свързаността на различните приложения, използвани от донорите и болничните служители
3. Golang Accounts Server - връзка с Accounts Microservice
4. Golang BloodBank Server - връзка с BloodBank Microservice
5. Database Connector – предназначението на модула е да осъществи връзка с базите данни, служеща за извлечане и въвеждане на информация.



3.3 Обосновка на проектирането

Ключовият принцип на microservices архитектурата е простотата. Приложението стават по-лесни за изграждане и поддръжка, когато са разделени на набор от по-малки, композируеми фрагменти. Управлението на кода също става по-малко болезнено, тъй като всеки микросървис въобще представлява отделна част от кода. Услугите могат да бъдат внедрени с помощта на различни програмни езици, бази данни и софтуерна среда. Това позволява всяка услуга да бъде внедрена, възстановена, преразпределена и управлявана независимо.

Микросървисите позволяват изграждане на продукти вместо проекти. Всъщност архитектурите на микросервисите канят екипи да се съсредоточат върху изграждането на бизнес функционалност, вместо да пишат “glue code”. С други думи, екипите за разработка са организирани около бизнес възможности, а не технологии. Това означава, че услугите са адаптивни за използване в множество контексти. Една и съща услуга може да бъде използвана повторно в повече от един бизнес процес или по различни бизнес канали в зависимост от нуждата. Всеки член на екипа е отговорен за определена услуга, която води до изграждане на интелигентен, многофункционален екип.

4 ДАННОВО ПРОЕКТИРАНЕ

4.1 Описание на данните

Данновото проектиране се разпределя на 3 части:

1. Проектиране на данните в Accounts Service
2. Проектиране на данните в BloodBank Service
3. Проектиране на данните в UI Service

Данните в Accounts Service се обработват чрез операции над 2 вида структури от данни, където Donor представя кръводарителите в системата, а Acceptor - пациентите нуждаещи се от кръв :

```
// Donor is a struct used to represent the first account type
// in LifeBlood system - blood donors
type Donor struct {
    ID string `json:"id"`
    FirstName string `json:"name"`
    LastName string `json:"lastName"`
    PhoneNumber string `json:"phone"`
    Email string `json:"email"`
    Age string `json:"age"`
    Gender string `json:"gender"`
    BloodGroup string `json:"bloodGroup"`
    City string `json:"city"`
    RegistrationDate string `json:"regDate"`
}

// Acceptor is a struct used to represent the second account
// type in LifeBlood system - blood acceptors
type Acceptor struct {
    ID string `json:"id"`
    FirstName string `json:"name"`
    LastName string `json:"lastName"`
    BloodGroup string `json:"bloodGroup"`
    City string `json:"city"`
    BloodCenter string `json:"bloodCenter"`
    RegistrationDate string `json:"regDate"`
}
```

Съхраняване на използваните в bloodbank-service MySQL база данни:

```
// Donation is a struct used to represent Donation
// entity in the LifeBlood system
type Donation struct {
    DonationID int `json:"donationId"`
    UserID string `json:"userId"`
    BloodType string `json:"bloodType"`
    BloodCenter string `json:"bloodCenter"`
    Amount string `json:"amount"`
    Date string `json:"date"`
    Status string `json:"status"`
}
```

Данните се обработват посредством MySql команди от back-end системата bloodbank-service.

4.2 Речник на данните

Функции използвани за агрегация на данните в bloodbank-service:

- func updateDonation(donation Donation) error
- func insertDonation(donation Donation) (int, error)
- func getDonationListByUserID(userID string) ([]Donation, error)
- func getDonationList(userID string) ([]Donation, error) {
- func removeDonation(donationID int) error

Функции използвани за агрегация на данните в accounts-service:

- func (app *App) getAllDonors(w http.ResponseWriter, r *http.Request)
- func (app *App) getAllAcceptors(w http.ResponseWriter, r *http.Request)
- func (app *App) getDonorByID(w http.ResponseWriter, r *http.Request)
- func (app *App) getAcceptorByID(w http.ResponseWriter, r *http.Request)
- func (app *App) updateDonorByID(w http.ResponseWriter, r *http.Request)
- func (app *App) updateAcceptorByID(w http.ResponseWriter, r *http.Request)
- func (app *App) getDonorsByBloodGroup(w http.ResponseWriter, r *http.Request)
- func (app *App) getAcceptorsByBloodGroup(w http.ResponseWriter, r *http.Request)
- func (app *App) addDonor(w http.ResponseWriter, r *http.Request)
- func (app *App) addAcceptor(w http.ResponseWriter, r *http.Request)
- func (app *App) deleteDonorByID(w http.ResponseWriter, r *http.Request)

Всяка една функция представлява метод на структурата данни App, с която се извършва релацията със получената заявка и базата.

```

// App is a wrapper struct over Router and Database used to manage db connections and
// endpoint requests centrally
type App struct {
    Router *mux.Router
    Database *sql.DB
}

// SetupRouter is used to provide mapping between different endpoints hit and handler
// functions
func (app *App) SetupRouter() {
    app.Router.
        Methods("GET").
        Path("/") .
        HandlerFunc(app.homePage)

    app.Router.
        Methods("GET").
        Path("/accounts/donors") .
        HandlerFunc(app.getAllDonors)

    app.Router.
        Methods("POST").
        Path("/accounts/donors") .
        HandlerFunc(app.addDonor)

    app.Router.
        Methods("POST").
        Path("/accounts/acceptors") .
        HandlerFunc(app.addAcceptor)

    app.Router.
        Methods("GET").
        Path("/accounts/donors/{id:[a-zA-Z0-9]+}") .
        HandlerFunc(app.getDonorByID)

    app.Router.
        Methods("DELETE", "OPTIONS").
        Path("/accounts/donors/{id:[a-zA-Z0-9]+}") .
        HandlerFunc(app.deleteDonorByID)

    app.Router.
        Methods("GET").
        Path("/accounts/acceptors/{id:[a-zA-Z0-9]+}") .
        HandlerFunc(app.getAcceptorByID)

    app.Router.
        Methods("DELETE", "OPTIONS").
        Path("/accounts/acceptors/{id:[a-zA-Z0-9]+}") .
        HandlerFunc(app.deleteAcceptorByID)

    app.Router.
        Methods("GET").
        Path("/accounts/acceptors") .
        HandlerFunc(app.getAllAcceptors)

    app.Router.
        Methods("PUT", "OPTIONS").
        Path("/accounts/donors/{id:[a-zA-Z0-9]+}") .
        HandlerFunc(app.updateDonorByID)

    app.Router.
        Methods("PUT", "OPTIONS").
        Path("/accounts/acceptors/{id:[a-zA-Z0-9]+}") .
        HandlerFunc(app.updateAcceptorByID)

    app.Router.
        Methods("GET").
        Path("/accounts/donors/bloodtype/{bloodGroup:[a-zA-Z0-9]+}") .
        HandlerFunc(app.getDonorsByBloodGroup)

    app.Router.
        Methods("GET").
        Path("/accounts/acceptors/bloodtype/{bloodGroup:[a-zA-Z0-9]+}") .
        HandlerFunc(app.getAcceptorsByBloodGroup)
}

```

5 КОМПОНЕНТНО ПРОЕКТИРАНЕ

Компоненти на системата:

1. Accounts Microservice

Accounts излага REST API имплементиран на Go предоставящ операции, които се извършват от болничните служители над кръводарителите и кръвонуждаещите се регистрирани в системата. Болничните служители могат да добавят, изтриват, редактират и потърсят по кръвна група регистрирани пациенти или донори.

Accounts използва контейнеризирана MySQL инстанция хостната в Docker като част от persistance layer-а на LifeBlood.

2. BloodBank Microservice

Bloodbank излага REST API имплементиран на GO предоставящ операции, за извлечане, добавяне, изтриване и промяна на данни за кръвни банки. Всяка кръвна банка се свързва с еднозначно с регистриран потребител, чрез идентификационен номер

Bloodbank използва MySQL инстанция като част от persistance layer-а на LifeBlood

3. LifeBlood UI

Потребителският интерфейс е реализиран чрез React JS Framework и уеб компоненти SAP UI5 Web Components. Уеб компонентите са стилизираны по стандартите на SAP и разработения от компания Fiori Design Language. По този начин се постига консистентност и по-добро потребителско изживяване.

Потребителският интерфейс използва Google Maps API за визуализацията на центровете за кръводаряване.

6 ПРОЕКТИРАНЕ НА ПОТРЕБИТЕЛСКИЯ ИНТЕРФЕЙС

6.1 Обобщение на потребителския интерфейс

Потребителският интерфейс предоставя възможност за регистрация към всеки, които би желал да извърши кръводаряване. След като създаде своя профил, потребителят има възможност да следи последните новини и статии по темата за кръводаряване, както и да вижда актуална статистика за брой дарители, спасени хора и центрове за кръводаряване.

Регистриран потребител може да заяви желание за кръводаряване към център по негов избор. След като изпрати заявка, центърът се свърза с него и си уговорят среща за първоначален медицински преглед и последващи стъпки към кръводаряването.

6.2 Екранни изображения

The screenshot shows the 'Login' screen of the LifeBlood application. The background is light red. At the top center, the 'LifeBlood' logo is displayed. Below it, the word 'Login' is centered. There are two input fields: one labeled 'Email' and another labeled 'Password', both with placeholder text. A red 'Login' button is located below the password field. At the bottom right of the white login area, there is a link 'Become a Donor'.

Примерна форма за влизане в системата чрез потребителско име (имейл) и парола.

LifeBlood

Become a Donor

✓ Personal Information

Full Name

Telephone

Age

Gender

Blood

✓ Account Information

Email

Password

Confirm Password

Register

Примерна форма за регистрация на Донор в системата

 LifeBlood

Home Donate Donations Donors Patients Map

Welcome, !

 9 210
Saved Lives

 16 498
Donors

 6
Blood Centers

News

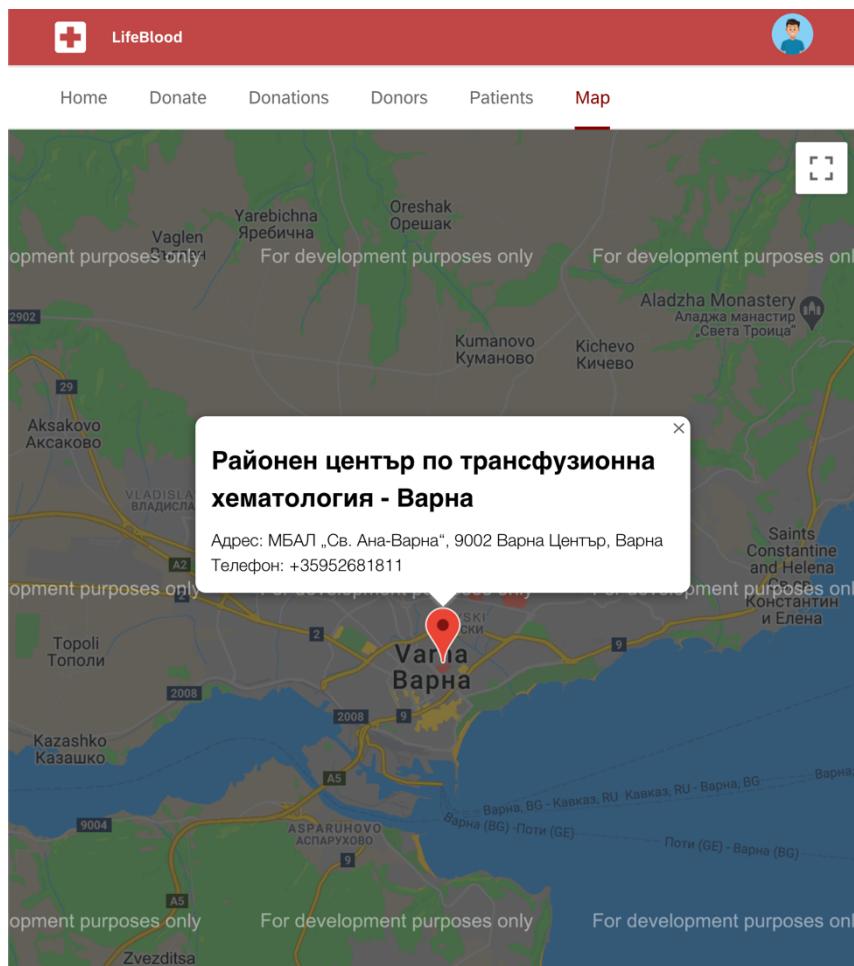
 Latest medical advice 21 December 2020

We're asking blood donors to keep donating as normal unless you have been in contact with, or are currently infected by, the virus. Please read the latest advice before attending.

 Mum urges blood donors to give blood over Christmas 18 December 2020

The mum of a little boy who will need a lifesaving blood transfusion just days before Christmas is backing an NHS call for donors to make and keep an appointment to give blood over the festive period.

Начален екран за болничен служител и донор



Google Maps с локация на болница за кръводаряване

7 МАТРИЦА НА ИЗИСКВАНИЯТА

<i>ID</i>	<i>User Interface</i>	<i>Accounts Service</i>	<i>Bloodbank Service</i>	<i>Accounts Database</i>	<i>Bloodbank Database</i>	<i>Firebase Database</i>
FR_1	1	1	0	1	0	0
FR_2	1	1	0	1	0	0
FR_3	1	0	1	0	1	1
FR_4	1	0	1	0	1	1
FR_5	1	0	0	0	0	0
FR_6	1	0	0	0	0	1
FR_7	1	0	0	0	0	1
FR_8	1	0	1	0	1	0
FR_9	1	0	1	0	1	0
FR_10	1	0	0	0	0	0
FR_11	1	1	0	1	0	0
FR_12	1	1	0	1	0	0
FR_13	1	0	0	0	0	0
FR_14	1	1	0	1	0	0
FR_15	1	1	0	1	0	0
FR_16	1	1	0	1	0	0
FR_17	1	0	0	0	0	0

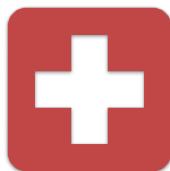
Легенда

0 - Неотговаряещ на изискване

1 - Отговаряящ на изискване

СИСТЕМА ЗА КРЪВОДАРЯВАНЕ

LIFEBLOOD



КУРСОВ ПРОЕКТ

ДИСЦИПЛИНА: ПРОЕКТИРАНЕ И ИНТЕГРИРАНЕ НА СОФТУЕРНИ СИСТЕМИ

ФАЗА 3: РЕАЛИЗАЦИЯ НА СИСТЕМАТА

ВЕРСИЯ 1.0

ФАК. №	ИМЕ НА СТУДЕНТ	СЕКЦИЯ ОТ ДОКУМЕНТА
62113	Момчил Игнатов	ВЪВЕДЕНИЕ ИЗПОЛЗВАНИ ТЕХНОЛОГИИ РЕАЛИЗАЦИЯ НА ПОТРЕБИТЕЛСКИ ИНТЕРФЕЙС;
62117	Гъокан Сюлейманов	РЕАЛИЗАЦИЯ НА ПОТРЕБИТЕЛСКИ ИНТЕРФЕЙС; РЕАЛИЗАЦИЯТА НА БАЗАТА ОТ ДАННИ; РЕАЛИЗАЦИЯ НА БИЗНЕС ЛОГИКАТА
62137	Илкнур Мустафа	РЕАЛИЗАЦИЯТА НА БАЗАТА ОТ ДАННИ; ВНЕДРЯВАНЕ НА СИСТЕМАТА; РАЗПРЕДЕЛЕНИЕ НА ДЕЙНОСТИТЕ ПО РЕАЛИЗАЦИЯТА; РЕАЛИЗАЦИЯ НА БИЗНЕС ЛОГИКАТА

ДЕКЕМВРИ, 2020

Съдържание

1	Въведение	3
1.1	Цел.....	3
1.2	Дефиниции и акроними	3
2	Използвани технологии.....	4
3	Реализация на базата от данни	5
4	Реализация на бизнес логиката.....	8
5	Реализация на потребителския интерфейс	12
6	Внедряване на системата.....	19
7	Разпределение на дейностите по реализацията	21

1 ВЪВЕДЕНИЕ

1.1 Цел

Настоящият документ описва методите за реализация и внедряване на софтуерната система.

1.2 Дефиниции и акроними

- **API (Application Programming Interface)** – приложно-програмен интерфейс
- **REST (REpresentational State Transfer)** – стил софтуерна архитектура за реализация на уеб услуги
- **Микросървис архитектура** – вариант на ориентираната към услуги архитектура, при която системата е разделена на няколко слабо свързани помежду си подсистеми (**микросървиси**)
- **Контейнер** - стандартна единица софтуер, който пакетира кода и всички негови зависимости, така че приложението да работи бързо и надеждно от една изчислителна среда в друга.

2 ИЗПОЛЗВАНИ ТЕХНОЛОГИИ

Системата има трислойна архитектура - потребителски интерфейс, бизнес логика и база данни.

Потребителският интерфейс е реализиран чрез React JS Framework и уеб компоненти SAP UI5 Web Components. Уеб компонентите са стилизиирани по стандартите на SAP и разработения от компания Fiori Design Language. По този начин се постига консистентност и по-добро потребителско изживяване.

Потребителският интерфейс използва Google Maps API за визуализацията на центровете за кръводаряване.

Bloodbank излага REST API имплементиран на GO предоставящ операции, за извлечане, добавяне, изтриване и промяна на данни за кръвни банки. Всяка кръвна банка се свързва с еднозначно с регистриран потребител, чрез идентификационен номер.

Bloodbank използва MySQL инстанция като част от persistance layer-а на LifeBlood на Go.

Accounts излага REST API имплементиран на Go предоставящ операции, които се извършват от болничните служители над кръводарителите и кръвонуждаещите се регистрирани в системата. Болничните служители могат да добавят, изтриват, редактират и потърсят по кръвна група регистрирани пациенти или донори.

Accounts използва контейнеризирана MySQL инстанция хостната в Docker като част от persistance layer-а на LifeBlood.

3 РЕАЛИЗАЦИЯ НА БАЗАТА ОТ ДАННИ

Системата *LifeBlood* е интегрирана да работи с три различни бази от данни.

- Контейнеризирана MySQL Accounts база данни, хостаната в Docker, съхраняваща различните типове акаунти в системата (донори и пациенти):

```
CREATE TABLE donors (
    id varchar(32) NOT NULL,
    name varchar(32),
    lastName varchar(32),
    phone varchar(32),
    email varchar(32),
    age integer,
    gender varchar(32),
    bloodGroup varchar(32),
    city varchar(50),
    regDate varchar(32),
    PRIMARY KEY (id);

CREATE TABLE acceptors (
    id varchar(32) NOT NULL,
    name varchar(32),
    lastName varchar(32),
    bloodGroup varchar(32),
    city varchar(50),
    bloodCenter varchar(250),
    regDate varchar(32),
    PRIMARY KEY (id));

INSERT INTO donors(id, name, lastName, phone, email, age, gender, bloodGroup, city, regDate)
VALUES ('12','Ivan','Petrov','08978654321','ivanp@abv.bg','31','MALE','AB','Sofia', 'Sun Mar 15
02:44:15 EET 2019');

INSERT INTO acceptors(id, name, lastName, bloodGroup, city, bloodCenter, regDate)
VALUES ('12','Ivan','Petrov','AB','Sofia', 'РЦ по трансфузионна хематология - Пловдив', 'Sun Mar
15 02:44:15 EET 2019');
```

- MySQL *bloodbankdb* база данни, съхраняваща данни за налични кръвни донори *donations*

```

CREATE DATABASE `bloodbankdb`;

CREATE TABLE `bloodbankdb`.`donations` (
`donationId` INT NOT NULL AUTO_INCREMENT,
`userId` VARCHAR(255) NOT NULL,
`bloodType` VARCHAR(255) NOT NULL,
`bloodCenter` VARCHAR(511) NOT NULL,
`amount` VARCHAR(255) NOT NULL,
`date` VARCHAR(255) NOT NULL,
`status` VARCHAR(255) NOT NULL,
PRIMARY KEY (`donationId`)
);

INSERT INTO `bloodbankdb`.`donations`(`userId`, `bloodType`,
`bloodCenter`, `amount`, `date`, `status`)
VALUES("Ivan-Ivanov@gmail.com", "AB", "РАЙОНЕН Ц-Р ПО
ТРАНСФУЗИОННА ХЕМАТОЛОГИЯ – Стара Загора", "310mil.", "23.2.2020",
"Completed");

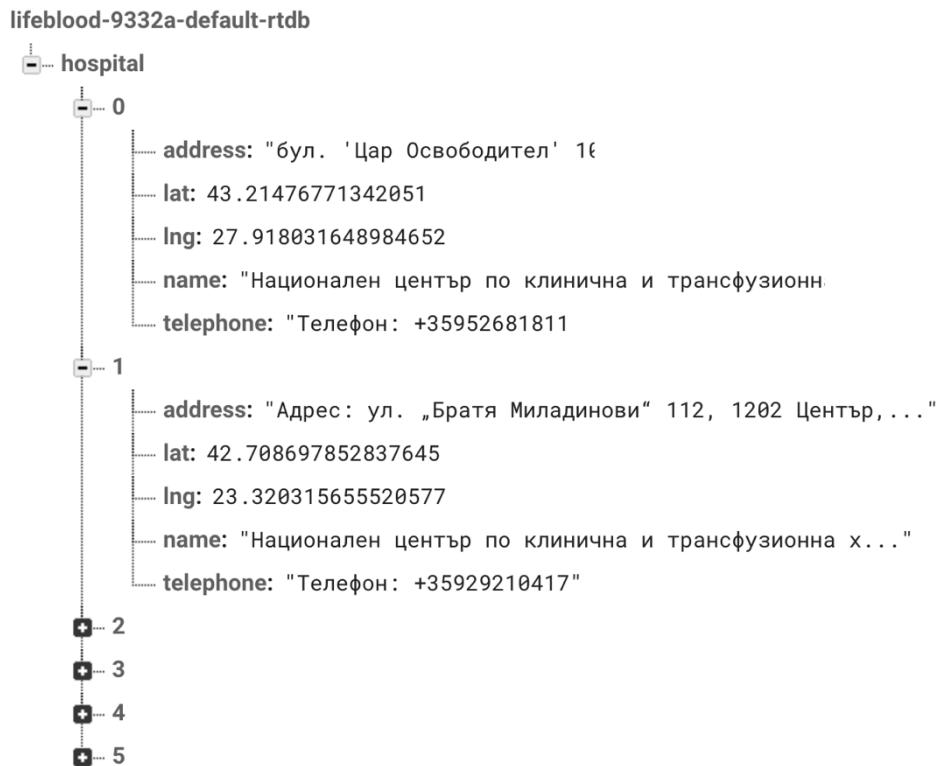
INSERT INTO `bloodbankdb`.`donations`(`userId`, `bloodType`,
`bloodCenter`, `amount`, `date`, `status`)

```

*donationId - уникален числовой идентификатор за всеки донор извършена
userId - идентификатор тип низ на текущ регистриран потребител
bloodType - тип кръвна група O, A, B или AB*

*bloodCenter - болнично заведение, в което се съхранява кръвната банка
amount - количество дарена кръв, ако кръводаряването е завършено
date - дата на извършване на кръводаряване
status - статус на кръводаряване - In Progress, Complete*

- *NoSql база данни Firebase, която съхранява данни за регистрирани болници в JSON формат:*



4 РЕАЛИЗАЦИЯ НА БИЗНЕС ЛОГИКАТА

Bloodbank service, осигурява REST API за работа с кръвни банки:

> **GET <bLoobank-server>/api/donations**

Връща JSON обект, съдържащ всички налични кръвни банки в системата:

> **GET <bLoobank-server>/api/donations?userId=<ID>**

Връща JSON обект, съдържащ информация за налични кръвни банки за конкретен потребител

> **POST <bLoobank-server>/api/donations**

Връща 200 OK, Добавя нов Donation обект в базата ако изпратените данните в тялото на заявката са валидни.

В противен случай връща грешка статус 400

> **PUT <bLoobank-server>/api/donations/<ID>**

Променя информация в системата за конкретен Donation обект идентифициран от ID в URL заявката.

> **GET <bLoobank-server>/api/donations/<ID>**

Връща JSON обект с конкретен Donation обект в системата.

> **DELETE <bLoobank-server>/api/donations/<ID>**

Изтрива Donation обект, идентифициран от ID в URL на заявката.

Accounts Service, осигурява REST API за работа с донори и пациенти:

GET	GET /accounts/donors
GET	GET /accounts/acceptors
GET	GET /accounts/donors/:id
GET	GET /accounts/acceptors/:id
PUT	PUT /accounts/donors/:id
PUT	PUT /accounts/acceptors/:id
GET	GET /accounts/donors/bloodtype/:bloodGroup
GET	GET /accounts/acceptors/bloodtype/:bloodGroup
POST	POST /accounts/donors
POST	POST /accounts/acceptors
DEL	DELETE /accounts/acceptors/:id
DEL	DELETE /accounts/donors/:id

Описание на заявките:

> ***GET /accounts/donors***

Връща JSON обект, съдържащ всички донори в базата.

Status 500: Internal Server Error

> ***GET /accounts/acceptors***

Връща JSON обект, съдържащ всички пациенти нуждаещи се от кръв в базата.

Status 500: Internal Server Error

> ***GET /accounts/donors/:id***

Връща JSON обект, репрезентиращ донор обекта от базата със съответното id.

Status 200: Success

Status 404: Not found

Status 500: Internal Server Error

> ***GET /accounts/acceptors/:id***

Връща JSON обект, репрезентиращ acceptor обекта от базата със съответното id.

Status 200: Success

Status 404: Not found

Status 500: Internal Server Error

> ***PUT /accounts/donors/:id***

Модифицира донор обект от базата със съответното id със данните подадени в тялото на заявката.

Status 200: Success

Status 404: Not found

Status 500: Internal Server Error

> ***PUT /accounts/acceptors/:id***

Модифицира acceptor обект от базата със съответното id със данните подадени в тялото на заявката.

Status 200: Success

Status 404: Not found

Status 500: Internal Server Error

> ***GET /accounts/donors/bloodtype/:bloodGroup***

Връща JSON обект, съдържащ всички донори в базата от съответната bloodGroup.

> ***GET /accounts/acceptors/bloodtype/:bloodGroup***

Връща JSON обект, съдържащ всички пациенти от съответната bloodGroup нуждаещи се от кръв в базата.

> ***POST /accounts/donors***

Добавя донор в базата от данни, чийто структура от данни се популира от подадените в тялото на заявката данни.

Status 201: Created

Status 500: Internal Server Error

> ***POST /accounts/acceptors***

Добавя acceptor в базата от данни, чийто структура от данни се популира от подадените в тялото на заявката данни.

Status 201: Created

Status 500: Internal Server Error

> ***DELETE accounts/donors/:id***

Изтрива донор от базата данни по подадено id.

Status 200: Success

Status 500: Internal Server Error

> ***DELETE accounts/acceptors/:id***

Изтрива acceptor от базата данни по подадено id.

Status 200: Success

Status 500: Internal Server Error

5 РЕАЛИЗАЦИЯ НА ПОТРЕБИТЕЛСКИЯ ИНТЕРФЕЙС

Логин страница - ако потребителят е предварително регистриран като Донор или има администраторски достъп като болничен служител.



Ако потребителят не е регистриран, това може да стане чрез “Become a Donor” форма

LifeBlood

Become a Donor

✓ Personal Information

Иван

Иванов

+359888061112

23

Female

A

София, ул. "Иван Вазов" 8, ет. 3 ап. 6

✓ Account Information

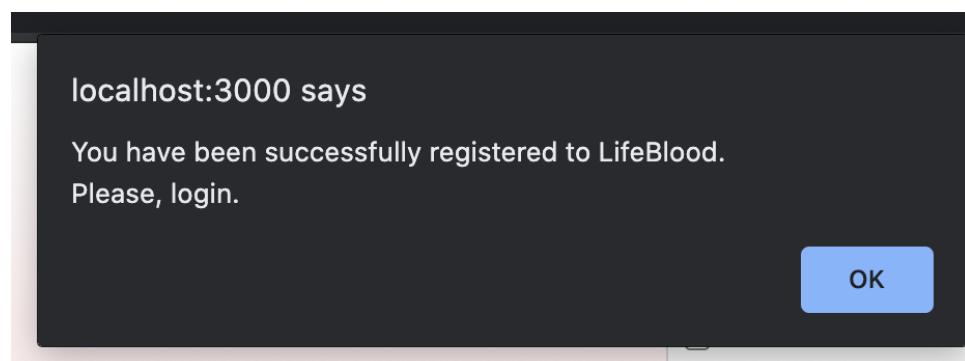
ivan.ivanov_199@abv.bg

.....

.....

Register

След успешна регистрация, потребителя получава съобщение да влезе в системата с регистрирания имейл и парола.



Начална страница - общ за болничен служител и краен потребител.

Потребителите виждат статистика на системата и последните новини

 LifeBlood 

Home Donate Map

Welcome, Иван!

 9 210
Saved Lives

 16 498
Donors

 6
Blood Centers

News

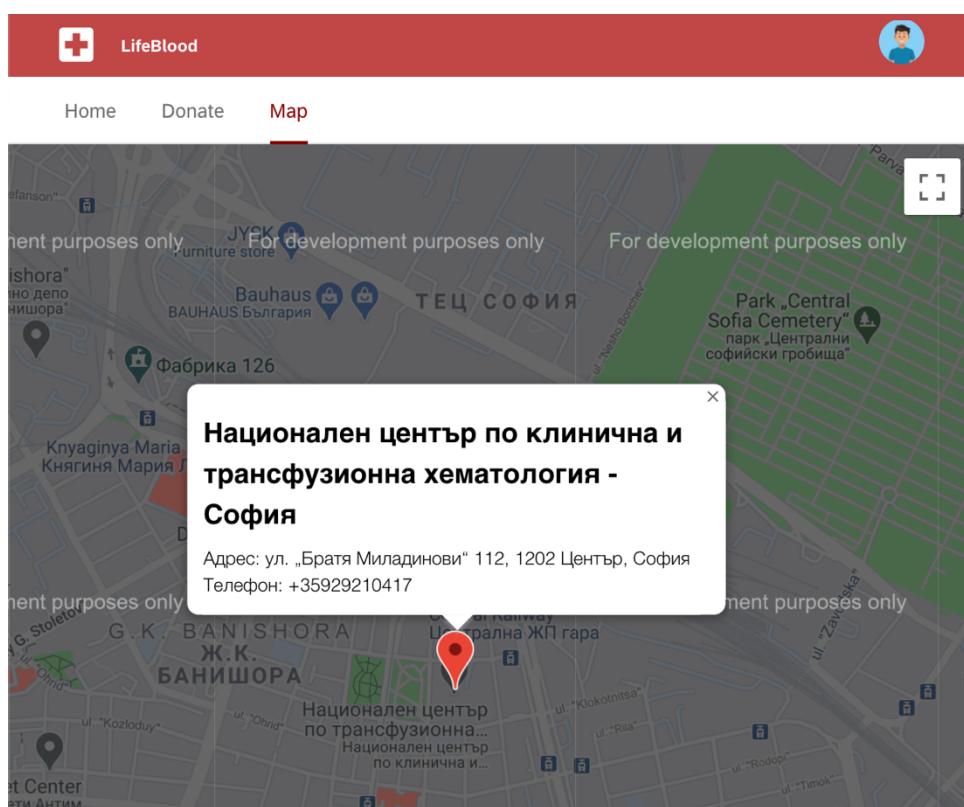
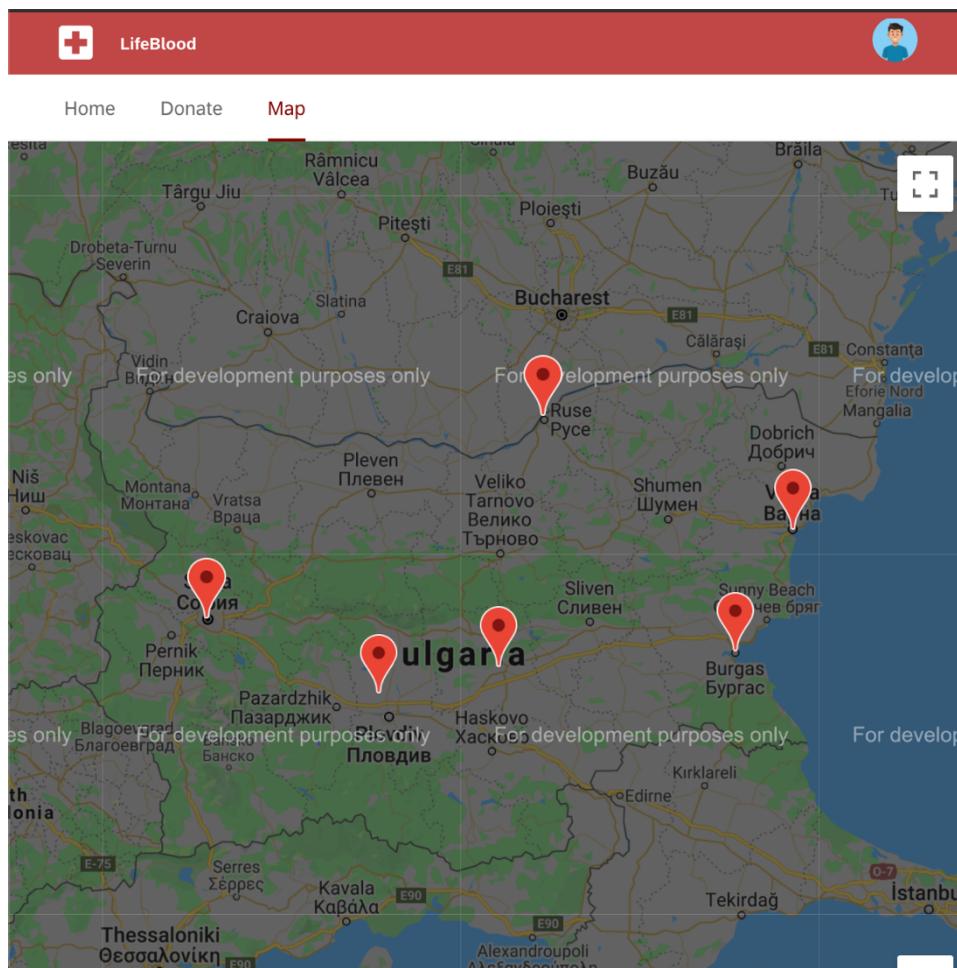
 Latest medical advice 21 December 2020

We're asking blood donors to keep donating as normal unless you have been in contact with, or are currently infected by, the virus. Please read the latest advice before attending.

 Mum urges blood donors to give blood over Christmas 18 December 2020

The mum of a little boy who will need a lifesaving blood transfusion just days before Christmas is backing an NHS call for donors to make and keep an appointment to give blood over the festive period.

Потребителя може да разглежда всички налични болници за кръводаряване, които работят с LifeBlood



След като е избрана конкретна болница, потребителят може да създаде заявка за кръводаряване:

The screenshot shows the LifeBlood mobile application. At the top, there is a dark header bar with the 'LifeBlood' logo and a user profile icon. Below the header, a navigation bar has three items: 'Home', 'Donate' (which is underlined in red), and 'Map'. The main content area is titled 'My Donations Иван' (My Donations Ivan). On the right side of this title is a red button with a white plus sign and the word 'Donate'. Below the title is a search bar with a placeholder 'Search' and a magnifying glass icon. Underneath the search bar is a table header with columns: Date, Blood Center, Amount, Status, and Actions. A message 'No donations found.' is displayed below the table. A modal window titled 'Donation Request' is open in the center. It contains a note: 'When you send the request, the blood center will contact you for appointment.' followed by a blue info icon. Below this is a question 'Where would you like to donate?' with a dropdown menu showing 'Национален център по клинична и трансфузионна хематология - София'. At the bottom of the modal are two buttons: a red 'Send' button and a white 'Cancel' button.

Потребителя вижда статус на своите кръводарявания

This screenshot shows the same 'My Donations Иван' screen from the previous image. The modal window is now closed. The table below the search bar displays one row of data: '17.1.2021 Национален център по клинична и трансфузионна хематология - София' in the 'Blood Center' column, '-' in the 'Amount' column, 'In Progress' in the 'Status' column, and a small trash can icon in the 'Actions' column.

Болничните служители от своя страна имат администраторски достъп до всички налични кръвни банки и могат да променят статуса на дарение след успешно кръводарение

ID	Date	Blood type	Blood Center	Amount	Status
1	23.2.2020	AB	РАЙОНЕН Ц-Р ПО ТРАНСФУЗИОННА ХЕМАТОЛОГИЯ - Стара Загора	400mil.	Completed
2	12.8.2020	AB	РАЙОНЕН Ц-Р ПО ТРАНСФУЗИОННА ХЕМАТОЛОГИЯ - Стара Загора	450mil.	Completed
4	5.3.2020	0	Районен център по трансфузионна хематология - Варна	300mil.	Completed
5	5.9.2020	0	Районен център по трансфузионна хематология - Варна	450mil.	Completed
6	15.1.2021	0	Районен център по трансфузионна хематология - Варна	312mil.	In Progress
18	12.8.2020	AB	РАЙОНЕН Ц-Р ПО ТРАНСФУЗИОННА ХЕМАТОЛОГИЯ - Стара Загора	450mil.	Completed
19	12.8.2020	AB	РАЙОНЕН Ц-Р ПО ТРАНСФУЗИОННА ХЕМАТОЛОГИЯ - Стара Загора	450mil.	Completed
20	16.1.2021	A	РЦ по трансфузионна хематология - Пловдив	300mil.	Completed
21	16.1.2021	A	Кръвен център - Бургас	450mil.	Completed
22	16.1.2021	A	РЦ по трансфузионна хематология - Пловдив	300mil.	Completed

Болничните служители виждат списък със информация за всички донори и пациенти.

Name	Last Name	Age	Gender	Blood Type	Telephone	City	Email	Actions
Petka	Petrova	31	MALE	B	08978654321	Sofia	ivanp@abv.bg	
Ivan	Petrov	31	Male	AB	08978654321	Sofia	ivanp_1998@abv.bg	
Иван	Иванов	23	Male	A	+359888061112	София, ул. "Иван Вазов" 8, ет. 3 ап. 6	ivan.ivanov_199@abv.bg	
Иван	Иванов	23	Male	A	+359888061112	София, ул. "Иван Вазов" 8, ет. 3 ап. 6	ivan.ivanov_199@abv.bg	

Name	Last Name	City	Blood Group	Blood Center	Actions
Ivan	Petrov	Sofia	AB	РЦ по трансфузионна хематология - Пловдив	
Ivaylo	Yosifov	Plovdiv	0	РЦ по трансфузионна хематология - Варна	

Болничните служители могат да търсят налична кръв по кръвна група, добавят нови нуждаещи се пациенти, да променят данни за всеки пациент и да изтрива пациенти.

Add Patient

Personal Information

Name

Last Name

City

Blood Group

Blood Center

Choose Blood Center

Add Cancel

6 ВНЕДРЯВАНЕ НА СИСТЕМАТА

Предварителни изисквания:

1. Golang Programming Language

Инсталиране и конфигурация : <https://golang.org/doc/install>

2. Docker Daemon

Инсталиране и конфигурация: <https://www.docker.com/products/docker-desktop>

3. NPM Package Manager

Инсталиране и конфигурация: <https://docs.npmjs.com/downloading-and-installing-node-js-and-npm>

4. Git Version Control System

Инсталиране и конфигурация: <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

5. MySQL WorkBench

Инсталиране и конфигурация:

<https://docs.bitnami.com/installer/infrastructure/lamp/configuration/configure-workbench/>

Внедряване:

1. Инсталлиране на трите микросървиси - <https://github.com/life-blood>

Помощни команди:

```
$ git clone git@github.com:life-blood/accounts-service.git  
$ git clone git@github.com:life-blood/lifeblood-ui.git  
$ git clone git@github.com:life-blood/bloodbank-service.git
```

2. Стартiranе на всеки микросървис

Навигирайте до root-а на всеки сървис в терминала ви и изпълнете следните команди за всеки един от тях. За целта ще ви трябват няколко прозорци в терминала.

- Accounts Service - уверете се, че Docker Desktop е стартиран

```
$ docker-compose up  
$ go run main.go
```

- BloodBank Service - уверете се, че MySQL Workbench е стартиран и конфигуриран правилно

```
$ go run main.go
```

- LifeBlood UI

```
$ npm install
```

```
$ npm run start
```

3. Отворете <http://localhost:3000/lifeblood-ui> във вашия уеб браузър.

7 РАЗПРЕДЕЛЕНИЕ НА ДЕЙНОСТИТЕ ПО РЕАЛИЗАЦИЯТА

Работата по проекта беше равномерно разпределена между участниците на екипа:

- Accounts Service (Имплементация и интеграция с UI): Илкнур Мустафа
- BloodBank Service (Имплементация и интеграция с UI): Гъокан Сюлейманов
- LifeBlood UI (Реализация): Момчил Игнатов